**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ**
**„ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”**

Методичні вказівки для лабораторних занять та самостійної роботи студентів з дисципліни

# ОСНОВИ СТВОРЕННЯ ГРАФІЧНИХ ДОДАТКІВ

Харків 2018

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ**
**„ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”**

Методичні вказівки для лабораторних занять та самостійної роботи студентів з дисципліни

# ОСНОВИ СТВОРЕННЯ ГРАФІЧНИХ ДОДАТКІВ

для студентів спеціальності 122 Комп'ютерні науки

Затверджено
редакційно-видавничою
радою університету,
протокол № 1 від 30.01.18 р.

Харків
НТУ «ХПІ»
2018

Методичні вказівки для лабораторних занять та самостійної роботи студентів з дисципліни «Основи створення графічних додатків» для студентів спеціальності 122 комп'ютерні науки / Уклад. Дашкевич А.О., Воронцова Д.В. – Харків : НТУ «ХПІ», 2018. – 13 с.

Укладачі А. О. Дашкевич
Д. В. Воронцова

Рецензент Л. М. Савченко

Кафедра геометричного моделювання та комп'ютерної графіки

# ВСТУП

Методичні вказівки для лабораторних занять містять теоретичний і практичний матеріал, який складає основу програмування засобами фреймворку Qt. В методичних вказівках розглядаються такі питання, як методи створення додатків з графічним інтерфейсом користувача з допомогою фреймворку Qt, а також тривимірних графічних додатків за стандартом OpenGL.

Паралельно з вивченням теоретичного матеріалу студентам необхідно набувати практичних навичок в галузі створення сучасних додатків з широким використанням графічних можливостей комп'ютерів. Завдання, які студенти виконують під керівництвом викладача та самостійно, спрямовані на розвиток логіки, творчого підходу у пошуках рішення, побудови алгоритмів, інтерфейсу користувача та навичок програмування.

В запропонованих методичних вказівках наведені структурні схеми побудови графічних додатків на основі фреймворку Qt і тривимірних графічних додатків засобами стандарту OpenGL. Пропонуються лістинг деяких програм, що допомагає сприйняттю та засвоєнню основних положень теоретичного матеріалу. Методичні вказівки містять опис 8 лабораторних занять.

ЛАБОРАТОРНЕ ЗАНЯТТЯ 1

**Основи створення графічного інтерфейсу користувача.**

Мета роботи: вивчення базових методів побудови інтерфейсу користувача з модулем Qt Widgets.

Завдання на роботу: створення текстового редактору з базовими функціями обробки тексту. В процесі виконання роботи необхідно ознайомитись з методами роботи в середовищі Qt Creator під час редагування форми і файлів програмного коду.

Порядок виконання роботи.

1. Створити новий проект “Додаток Qt Widgets”.

2. Створити наступні пункти меню:

- File з підпунктами Clear, Open, Save та Quit;
- Color з підпунктами Text (з підпунктами Black, Red, Green, Blue) та Background (з підпунктами White, Gray, Red, Green);
- Text effects з підпунктами To upper та To lower.

3. Для кожного пункту створених дій actionOpen, actionSave та ін. виконати операцію “Перейти до слоту” triggered().

4. Додати на форму основні віджети для реалізації можливостей додатку:

- QTextEdit для редагування тексту;
- кнопки QPushButton для реалізації дій користувача “Вихід з додатку” та “Очистити текст”;
- чекбокс QCheckBox для реалізації зміни регістру введеного тексту.

5. Об’явити та проініціалізувати об’єкт класу QTextEdit.

6. Для реалізації роботи кнопок QPushButton необхідно на кожній з них в редакторі форми клікнути правою кнопкою миші та перейти до слоту clicked().

7. Реалізація дій за кліками по кнопках:

- очищення текстового поля від введеного тексту;
- зберігання набраного тексту;
- завантаження файлу з текстом у вікно редактора.

8. Організація зміни регістру введеного тексту.

9. Зміна кольору тексту.

10. Організація обробки натискань кнопок клавіатури засобами класу QKeyEvent.

ЛАБОРАТОРНЕ ЗАНЯТТЯ 2

## Основи графічної системи фреймворку Qt. QPainter API

Мета роботи: набуття практичних навичок використання класу QPainter для реалізації операцій малювання на віджетах.

Завдання на роботу: створити додаток для побудови графіку функцій з інтерфейсом користувача для можливості встановлення та зміни коефіцієнтів функції, масштабування, візуального оформлення.

Порядок виконання роботи.

1. Основи малювання на QWidget. Створити новий проект, базовий клас - QWidget.

2. Малювання на QLabel. Додати на форму поле для виведення графічної інформації QLabel, видалити з нього текст. Додати на форму кнопку QPushButton. В процесі малювання слід ознайомитись з такими функціями:

- QPainter::drawEllipse();
- QPainter::drawLines();
- QPainter::drawPoint();
- QPainter::drawRect();
- QPainter::drawText();
- QPainter::drawRoundRect();

та з класами:

- QPoint[F];
- QLine[F];
- Qrect[F].

Для малювання різних об’єктів використовувати різні кольори та товщини ліній.

3. Малювання замкнених контурів та множин об’єктів з функціями:

- QPainter::drawPolygon() для малювання багатокутників на основі масиву QPoint, при цьому остання точка в масиві з’єднується з першою, таким чином формуючи замкнений багатокутник;
- QPainter::drawConvexPolygon() для малювання опуклих багатокутників, більш швидка, аніж QPainter::drawPolygon(), але якщо точки, що малюються, не складають опуклий багатокутник, то результат цієї функції не визначений;
- QPainter::drawPath() для малювання довільних форм, в тому числі кривих.

4. Заливка контурів обраним кольором.

5. Афінні перетворення з використанням функцій:

- QPainter::translate() - паралельне перенесення;
- QPainter::rotate() - поворот;
- QPainter::scale() - масштабування;
- QPainter::shear() - зсув.

6. Побудова графіку математичної функції.

7. Перетворення Window-Viewport.

8. Зміна кольорів ліній та фону. Виконати аналогічні дії для реалізації встановлення кольорів ліній сітки, синусоїди та фону.

9. Встановлення значення кроку сітки. Провести аналогічні дії для реалізації зміни кількості точок синусоїди..

10. Доповнити код для можливості малювання графіку вигляду

$A \cdot \sin(B \cdot x)$,

в якому користувач може задавати коефіцієнти A та B вручну з допомогою спінбоксів, текстових полів або слайдерів.

11. Реалізувати можливість зміни товщини ліній графіку, осей, сітки з використанням будь-яких придатних віджетів.

12. Реалізувати можливість збереження отриманого графіку в файл з використанням функції QPixmap::save(). Обов’язково використати вивчену функцію QFileDialog для задання імені файлу.

13. Отримана синусоїда перевернута догори ногами внаслідок особливостей координатної системи Qt. Змінити код функції drawSin() для того, щоб синусоїда відображалась математично коректно.

## ЛАБОРАТОРНЕ ЗАНЯТТЯ 3

## Графічна підсистема Qt і клас QGraphicsScene. Малювання ліній з використанням миші

Мета роботи: набуття практичних навичок побудови складних графічних додатків на основі класу QGraphicsScene.

Завдання на роботу: створити додаток для малювання і редагування геометричних примітивів: точок, ліній, трикутників, овалів та ін. з інтерфейсом користувача.

Порядок виконання роботи.

1. Створити новий проект на основі класу QWidget або QMainWindow.

2. На форму додати віджети Graphics View для візуалізації

намальованих об’єктів і необхідну кількість PushButton для роботи додатку.

3. Додати в проект новий клас, успадкований від QGraphicsScene.

4. Реалізувати програмний код обробки подій миші з використанням методів mousePressEvent(QGraphicsSceneMouseEvent *e) для визначення натискання кнопки миші, mouseMoveEvent(QGraphicsSceneMouseEvent *e) для визначення руху миші та mouseReleaseEvent(QGraphicsSceneMouseEvent *e) для визначення моменту закінчення натискання.

5. Реалізація програмного коду малювання прямої. Малювання здійснюватиметься тільки за умови натискання на відповідну QPushButton,

6. Створити список для зберігання намальованих ліній.

7. Реалізація малювання довільної кількості прямих.

8. Реалізація програмного коду малювання еліпсів та прямокутників.

9. Реалізація програмного коду малювання трикутників.

10. Реалізація програмного коду малювання багатокутників з можливістю задання кількості кутів користувачем.

11. Реалізація програмного коду відображення контурів об’єктів, що малюються.

12. Реалізація ефекту малювання пензлем плавних кривих ліній.

13. Реалізація можливості зміни кольорів ліній, фону, товщини та типу ліній.

14. Реалізація можливості виділення намальованих об’єктів прямокутником.

15. Реалізація видалення об’єктів, що були виділені.

16. Реалізація збереження намальованої сцени в файл зображення (формат файлу .jpg або .bmp) з використанням методу QGraphicsScene::render().

17. Додати можливості з малювання тексту.

## ЛАБОРАТОРНЕ ЗАНЯТТЯ 4

## **Основи роботи з зображеннями. Клас QImage**

Мета роботи: вивчення методів класу QImage.

Завдання на роботу: створити додаток для обробки цифрових зображень.

Порядок виконання роботи

1. Створити новий проект, базовий клас QWidget або QMainWindow.

2. На форму додати два віджети Graphics View для вхідного і

обробленого зображень і групу Push Button для реалізації наступних ефектів:

- завантаження зображення;
- переведення кольорового зображення у відтінки сірого кольору;
- пікселізація;
- закручування зображення в спіраль;
- збереження обробленого зображення.

Розміри Graphics View задати кратними до 10 з співвідношенням ширини до висоти як 4:3, наприклад, (400×300).

3. Завантаження і збереження зображень здійснювати з використанням QFileDialog.

4. Реалізація переводу кольорового зображення у відтінки сірого.

5. Реалізація ефекту пікселізації – розбити зображення на квадрати, наприклад, 10x10 пікселів. В кожному квадраті розрахувати середнє значення кольорів усіх пікселів для трьох каналів: червоного, зеленого та синього. Для цього реалізувати функцію meanPixel(). Розраховане середнє значення кольору присвоїти усім пікселям поточного квадрату.

6. Реалізація ефекту закручування зображення в спіраль.

7. Реалізація геометричних перетворень зображення з використанням функцій:

- QMatrix::translate(double tx, double ty);
- QMatrix::rotate(double angle);
- QMatrix::scale(double sx, double sy);
- QMatrix::shear(double sh, double sv).

Самостійна робота.

1. Додати можливість зміни розміру квадрату для ефекту пікселізації.

2. Реалізація можливості зміни кута та центру закручування.

3. Анімація ефекту закручування з використанням таймеру для кутів від 0 до 360 градусів.

## ЛАБОРАТОРНЕ ЗАНЯТТЯ 5

## **Основи QML та Qt Quick. Анімація та графіка в додатках Qt Quick**

Мета роботи: вивчення основ мови QML та створення додатків QtQuick, вивчення і робота зі станами та анімацією. Малювання в додатках Qt Quick.

Завдання на роботу: створити макет графічного інтерфейсу додатку для малювання і обробки цифрових зображень. Розробити анімаційні ефекти для елементів інтерфейсу додатку для малювання та обробки цифрових зображень. Реалізувати можливості з малювання та роботи з зображеннями.

Порядок виконання роботи.

1. Створити новий проект, тип проекту “Пустий додаток Qt Quick”.

2. Можливий вигляд макету інтерфейсу додатку представлено на рис. 5.1. Згідно наведеного на рис. макету розробити відповідний QML-код.

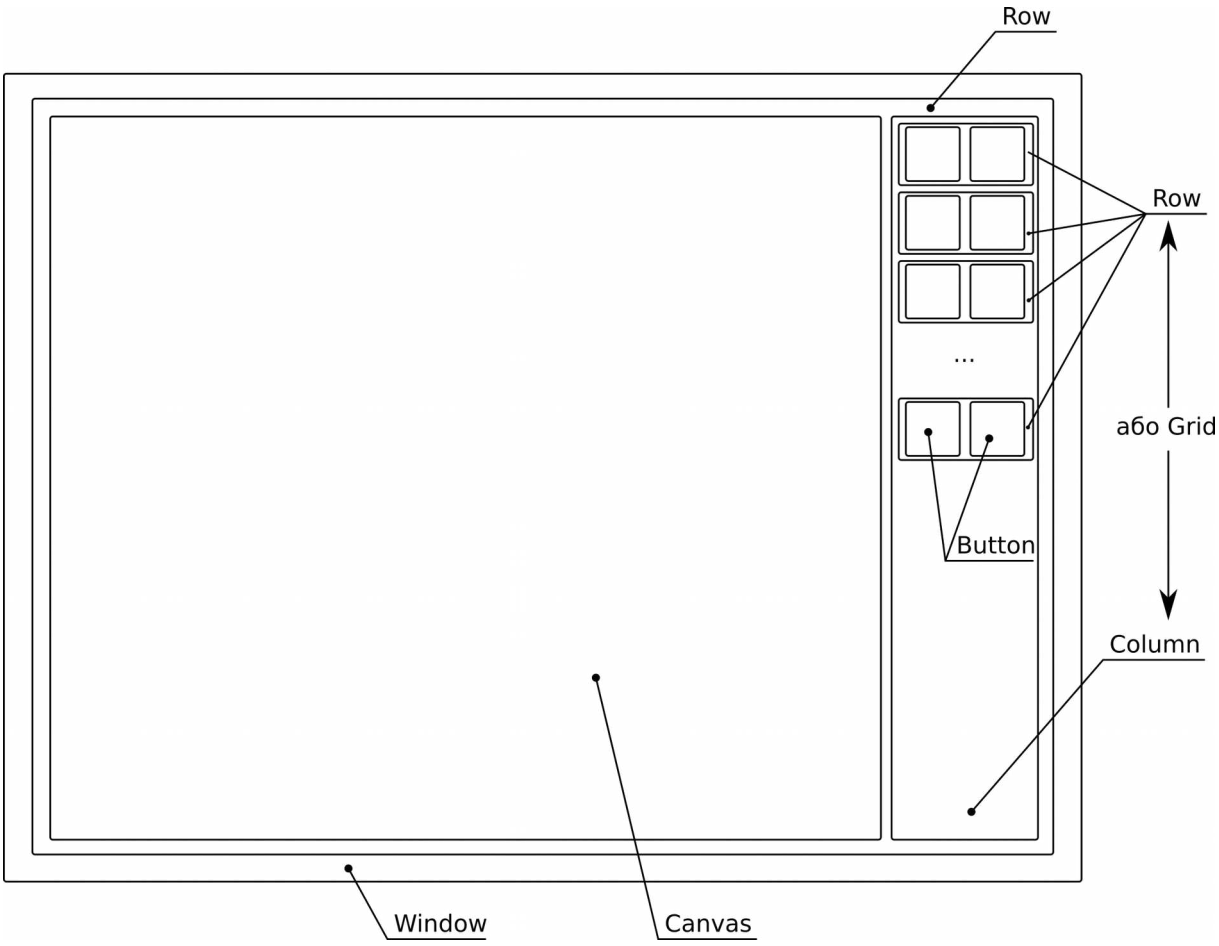


Рис. 5.1. Макет інтерфейсу додатку

3. Реалізація класу кнопки та надання йому відповідної функціональності.

4. Створити для типу Button два стани, наприклад:

- NORMAL – курсор не наведено на кнопку;
- HOVERED – курсор наведено на кнопку.

5. Додамо на форму кнопку для зміни кольору малювання:

6. Створення діалогу вибору кольору малювання.

7. Анімація властивостей. Створимо анімацію кнопки вибору кольорів після виклику діалогу ColorDialog.

8. Створення поля-полотна для малювання.

9. Додати на форму кнопки для зміни кольору фону та товщини ліній з ефектами анімації.

10. Намалювати на полотні графічні примітиви, задані в типі Context2D, використовуючи матеріал лекції та довідкову інформацію Qt.

11. Реалізація малювання олівцем та glow-ефекту.

12. Створити власний QML-тип, аналогічний до віджету QSpinBox. Створений тип розмістити в файлі SpinBox.qml. Розмістити достатню кількість елементів SpinBox на формі та реалізувати зміну величини shadowBlur для glow-ефекту. Також замінити кнопки зміни товщини ліній на SpinBox.

13. Додати на форму кнопку для включення glow-ефекту та реалізувати логіку її роботи.

14. Реалізація зберігання намальованого зображення в файл.

15. Додати для базового типу Button новий стан “PRESSED” та атрибут “previousColor” для реалізації коректної зміни кольорів фону кнопки під час зміни її станів.

## ЛАБОРАТОРНЕ ЗАНЯТТЯ 6

### Системи часток та спрайтова анімація в QML-додатках

Мета роботи: створення симуляції з системою часток і використання спрайтової анімації для комп’ютерної гри.

Завдання на роботу: розробити ефекти часток для реалізації поведінки об’єктів комп’ютерної гри, розробити спрайтову анімацію для головного персонажу гри.

Порядок виконання роботи.

1. Створюємо новий проект, тип проекту - “Додаток Qt Quick”.

2. Додамо в файл ресурсів “qml.qrc” файл-зображення частки.

3. Розмістимо в якості кореневого елемента проекту елемент Rectangle або Window.

4. Додамо елементи *Emitter*, який буде створювати частки, *ImageParticle* та *ParticleSystem* для можливості подальшої взаємодії з частками.

5. Реалізація обертання часток

6. Задання напряму руху часток.

7. Зміна траєкторії часток.

8. Намалювати різні варіанти часток та використати їх в симуляціях. Розміри зображень – від 8×8 до 16×16. Фон зображенням встановлювати прозорий.

## ЛАБОРАТОРНЕ ЗАНЯТТЯ 7

## Створення гри на QML

Мета роботи: створення комп'ютерної гри з використанням ефектів часток та спрайтової анімації.

Завдання на роботу: розробити програмну реалізацію поведінки головного персонажу та інших об'єктів комп'ютерної гри, розробити спрайтову анімацію для головного персонажу і об'єктів гри.

Порядок виконання роботи.

1. Створення ігрового персонажу - ракети. В якості головного героя розмістимо на формі елемент *Image*, який містить зображення ракети в форматі PNG з прозорим фоном.

2. Обертання персонажу.

3. Створення вихлопного полум'я ракети.

4. Створення ігрових об'єктів.

5. Реалізація ігрового циклу з використанням таймеру.

6. Реалізація перевірки зіткнень персонажу з об'єктами та з границями ігрового поля.

7. Спрайтова анімація об'єктів гри.

8. Моделювання вибуху персонажу засобами системи часток.

## ЛАБОРАТОРНЕ ЗАНЯТТЯ 8

## Шейдерні ефекти та тривимірна графіка за стандартом OpenGL

Мета роботи: створення додатку для візуалізації та анімації тривимірних геометричних об'єктів.

Завдання на роботу: розробити програмну реалізацію відображення, геометричних перетворень, текстурування та анімації геометричних об'єктів засобами стандарту OpenGL.

Порядок виконання роботи.

1. Завантаження масиву точок у відеокарту.

2. Розробка коду вершинного шейдеру для формування масиву вершин геометричного об’єкту.

3. Розробка коду фрагментного шейдеру і робота з атрибутами для задання кольорів для вершин фігури.

4. Геометричні перетворення в OpenGL ES 2.0. Створити модельну, видову та проекційні матриці з використанням класу QMatrix4x4.

5. Реалізація переміщень об’єкту з використанням клавіатури.

6. Реалізація текстурування об’єкту. Встановлюємо додаткові атрибути для текстурних координат.

7. Завантаження зображення, яке виступатиме в якості текстури.

8. Реалізація текстурування в коді шейдерів.

9. Анімація об’єктів з використанням таймеру.

Самостійна робота.

1. Робота режимами відображення: точки, лінії, смуги ліній, кільця з ліній, трикутники, смуги трикутників, віяла трикутників.

2. Намалювати прямокутник з використанням функцій glDrawArrays() та glDrawElements(), порівняти ці два варіанти відображення.

3. Задати функції для розрахування і формування масивів координат вершин правильних багатогранників: куб, тетраедр, октаедр, ікосаедр.

4. Реалізація переміщення камери з використанням миші: обертання камери навколо сцени з використанням події mouseMoveEvent і наближення-віддалення камери від сцени з використанням події wheelEvent.

5. Реалізація обертання заданого об’єкту навколо вісі Y за натисканням кнопок “A” та ”D”.

6. Перетворити код вершинного шейдеру так, щоб матриця перетворень розраховувалась в ньому, а не на центральному процесорі, для цього модельну, видову і проекційні матриці необхідно передати до шейдеру і здійснити їх перемноження в правильному порядку.

7. Додати на сцену правильні багатогранники у випадкових місцях з використанням циклу і зміни модельної матриці.

8. Реалізація текстурування усіх об’єктів сцени.

Навчальне видання

Методичні вказівки для лабораторних занять та самостійної роботи студентів з дисципліни

**Основи створення графічних додатків**

для студентів спеціальності 122 Комп'ютерні науки

Укладачі ДАШКЕВИЧ Андрій Олександрович
ВОРОНЦОВА Дар’я Володимирівна

Роботу к виданню рекомендувала проф. Шоман О. В.

В авторській редакції

План 2018 р., поз. 42
Підписано до друку 15.03.2019 р. Гарнітура Таймс. Ум. друк. арк. 0,8.

---

Видавничий центр НТУ “ХПІ”.
Свідоцтво про державну реєстрацію ДК № 5478 від 21.08.2017 р.
61002, Харків, вул. Кирпичова, 2

---

Самостійне електронне видання